

**Operating System and Design (19CS2106A)**  
**Advanced Lab- 1**

**Xv6 design, implementation, and customization.**

**1. add-a-user-program-to-xv6**

Clone xv6 in putty using the following link  
git clone [git://github.com/mit-pdos/xv6-public](https://github.com/mit-pdos/xv6-public).git xv6

nano filename.c

Add a program and exit from nano editor

The Makefile needs to be edited to make our program available for the xv6 source code for compilation.

nano Makefile

Add filename in Makefile in uprogs and extras

make clean

make

make qemu-nox

**2. printf.c**

```
#include "types.h"
```

```
#include "stat.h"
```

```
#include "user.h"
```

```
static void
```

```
putc(int fd, char c)
```

```
{
```

```
    write(fd, &c, 1);
```

```
}
```

```
static void
```

```
printint(int fd, int xx, int base, int sgn)
```

```
{
```

```
    static char digits[] = "0123456789ABCDEF";
```

```
    char buf[16];
```

```
    int i, neg;
```

```
    uint x;
```

```
    neg = 0;
```

```
    if(sgn && xx < 0){
```

```
        neg = 1;
```

```
    x = -xx;
} else {
    x = xx;
}
```

```
i = 0;
do{
    buf[i++] = digits[x % base];
}while((x /= base) != 0);
if(neg)
    buf[i++] = '-';

while(--i >= 0)
    putc(fd, buf[i]);
}
```

// Print to the given fd. Only understands %d, %x, %p, %s.

void

printf(int fd, const char \*fmt, ...)

```
{
    char *s;
    int c, i, state;
    uint *ap;


    state = 0;
    ap = (uint*)(void*)&fmt + 1;
    for(i = 0; fmt[i]; i++){
        c = fmt[i] & 0xff;
        if(state == 0){
            if(c == '%'){
                state = '%';
            } else {
                putc(fd, c);
            }
        } else if(state == '%'){
            if(c == 'd'){
                printint(fd, *ap, 10, 1);
                ap++;
            } else if(c == 'x' || c == 'p'){
                printint(fd, *ap, 16, 0);
                ap++;
            } else if(c == 's'){
                s = (char*)*ap;
                ap++;
                if(s == 0)
                    s = "(null)";
                while(*s != 0){
```

```

    putc(fd, *s);
    s++;
}
} else if(c == 'c'){
    putc(fd, *ap);
    ap++;
} else if(c == '%'){
    putc(fd, c);
} else {
    // Unknown % sequence. Print it to draw attention.
    putc(fd, '%');
    putc(fd, c);
}
state = 0;
}
}
}

```

OUTPUT

 osd-190031187@team-osd:~/xv6-public

```

SeaBIOS (version 1.11.0-2.el7)

iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980

Booting from Hard Disk...
xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap sta8
init: starting sh
190031187$ ls
.          1 1 512
..         1 1 512
README    2 2 2286
cat        2 3 14568
echo       2 4 13428
forktest  2 5 8256
grep       2 6 16104
init       2 7 14316
kill       2 8 13456
ln         2 9 13400
ls         2 10 16256
mkdir     2 11 13488
rm         2 12 13464
sh         2 13 24904
stressfs  2 14 14412
usertests  2 15 67312
wc         2 16 15236
zombie    2 17 13124
ps         2 18 14148
bt         2 19 13692
touch     2 20 13648
alarmtest  2 21 13860
touchex   2 22 13652
tail       2 23 18348
printf     2 24 12520
console    3 25 0
190031187$ printf
pid 4 printf: trap 14 err 5 on cpu 1 eip 0x39 addr 0x15d9--kill proc
190031187$ █

```

## UNIX system programming

### Compile-Link in GCC

Compile any program u like in gcc

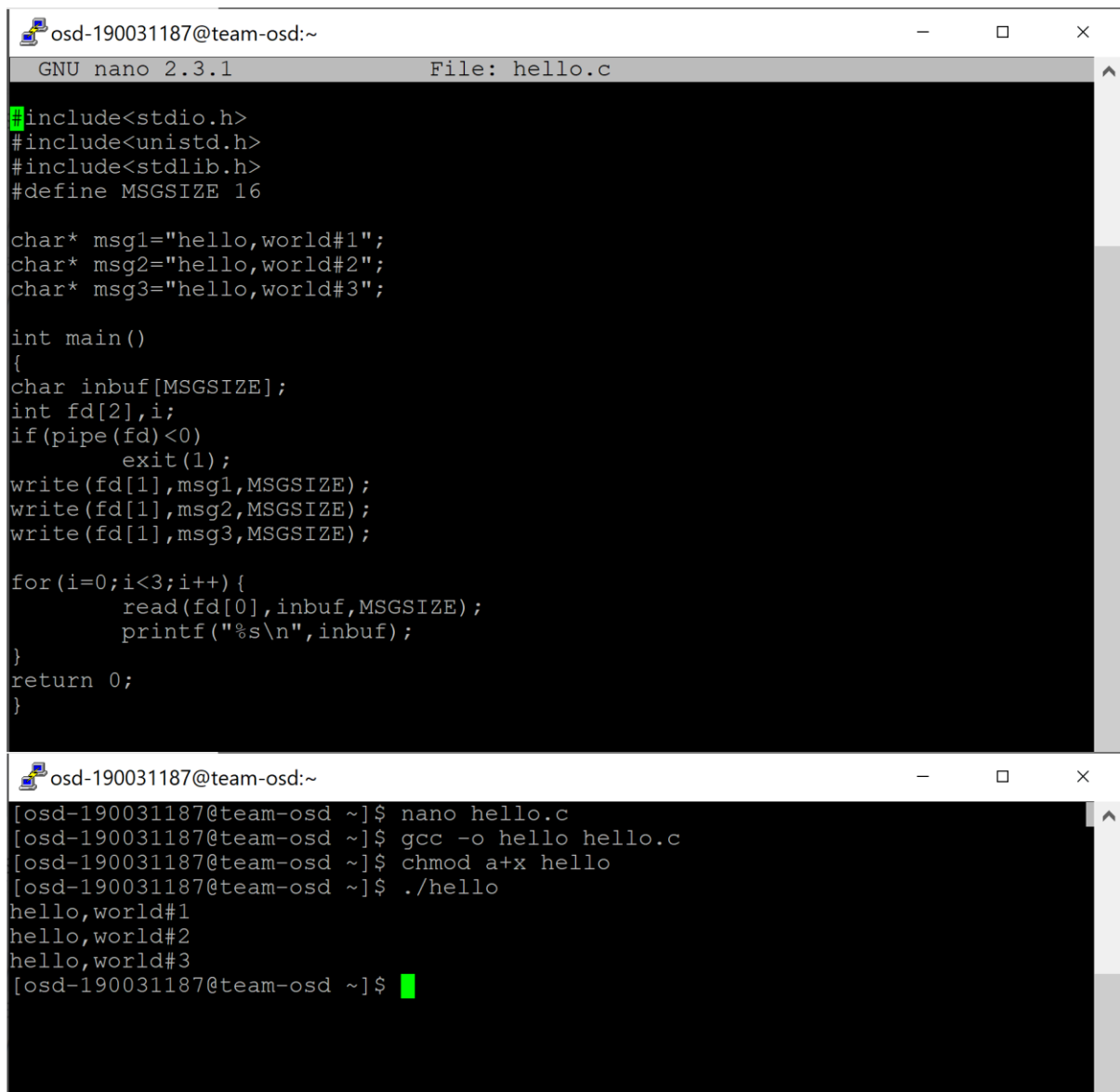
nano filename.c

Add code and exit

gcc filename.c

./a.out

Then link the code with another filename which doesnot exist in system using  
link filename1.c filename2.c



```
osd-190031187@team-osd:~  
GNU nano 2.3.1 File: hello.c  
#include<stdio.h>  
#include<unistd.h>  
#include<stdlib.h>  
#define MSGSIZE 16  
  
char* msg1="hello,world#1";  
char* msg2="hello,world#2";  
char* msg3="hello,world#3";  
  
int main()  
{  
char inbuf[MSGSIZE];  
int fd[2],i;  
if(pipe(fd)<0)  
    exit(1);  
write(fd[1],msg1,MSGSIZE);  
write(fd[1],msg2,MSGSIZE);  
write(fd[1],msg3,MSGSIZE);  
  
for(i=0;i<3;i++){  
    read(fd[0],inbuf,MSGSIZE);  
    printf("%s\n",inbuf);  
}  
return 0;  
}  
  
osd-190031187@team-osd:~  
[osd-190031187@team-osd ~]$ nano hello.c  
[osd-190031187@team-osd ~]$ gcc -o hello hello.c  
[osd-190031187@team-osd ~]$ chmod a+x hello  
[osd-190031187@team-osd ~]$ ./hello  
hello,world#1  
hello,world#2  
hello,world#3  
[osd-190031187@team-osd ~]$
```

## Static vs Dynamic Linking

### Static Linking:

When we click the .exe (executable) file of the program and it starts running, all the necessary contents of the binary file have been loaded into the process's virtual address space. However, most programs also need to run functions from the system libraries, and these library functions also need to be loaded. In the simplest case, the necessary library functions are embedded directly in the program's executable binary file. Such a program is statically linked to its libraries, and statically linked executable codes can commence running as soon as they are loaded.

### Dynamic Linking:

Every dynamically linked program contains a small, statically linked function that is called when the program starts. This static function only maps the link library into memory and runs the code that the function contains. The link library determines what are all the dynamic libraries which the program requires along with the names of the variables and functions needed from those libraries by reading the information contained in sections of the library. After which it maps the libraries into the middle of virtual memory and resolves the references to the symbols contained in those libraries. We don't know where in the memory these shared libraries are actually mapped: They are compiled into position-independent code (PIC), that can run at any address in memory

### Contents of a.out File

not so sure for this answer

If I'm not wrong use

objdump -i filename to see contents of a.out file

```
osd-190031187@team-osd:~
a.out:      file format elf64-x86-64

Disassembly of section .interp:

0000000000400238 <.interp>:
400238:      2f                (bad)
400239:      6c                insb    (%dx),%es:(%rdi)
40023a:      69 62 36 34 2f 6c 64  imul    $0x646c2f34,0x36(%rdx),%esp
400241:      2d 6c 69 6e 75      sub     $0x756e696c,%eax
400246:      78 2d             js      400275 <init-0x24b>
400248:      78 38             js      400282 <init-0x23e>
40024a:      36 2d 36 34 2e 73  ss sub  $0x732e3436,%eax
400250:      6f                outsl   %ds:(%rsi),(%dx)
400251:      2e 32 00          xor     %cs:(%rax),%al

Disassembly of section .note.ABI-tag:

0000000000400254 <.note.ABI-tag>:
400254:      04 00             add     $0x0,%al
400256:      00 00             add     %al,(%rax)
400258:      10 00             adc     %al,(%rax)
40025a:      00 00             add     %al,(%rax)
40025c:      01 00             add     %eax,(%rax)
40025e:      00 00             add     %al,(%rax)
400260:      47                rex.RXB
400261:      4e 55             rex.WRX push %rbp
400263:      00 00             add     %al,(%rax)
400265:      00 00             add     %al,(%rax)
400267:      00 02             add     %al,(%rdx)
400269:      00 00             add     %al,(%rax)
40026b:      00 06             add     %al,(%rsi)
40026d:      00 00             add     %al,(%rax)
40026f:      00 20             add     %ah,(%rax)
400271:      00 00             add     %al,(%rax)
...

Disassembly of section .note.gnu.build-id:

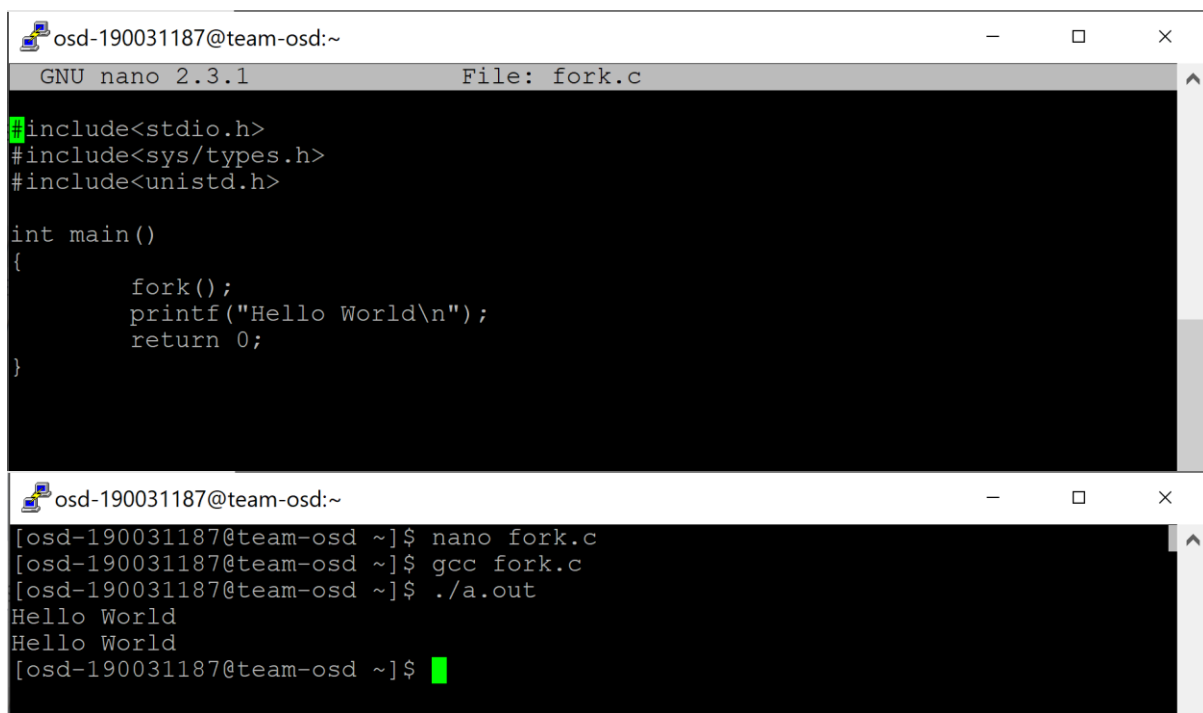
0000000000400274 <.note.gnu.build-id>:
400274:      04 00             add     $0x0,%al
400276:      00 00             add     %al,(%rax)
400278:      14 00             adc     $0x0,%al
40027a:      00 00             add     %al,(%rax)
```

**Program Execution and termination.**

```
nano filename.c  
Add code and exit  
gcc filename.c  
./a.out
```

We have various ways to terminate a process

```
Cntrl+z (or)  
killall gcc (or)  
kill -s SIGINT %1  
exit()
```



```
osd-190031187@team-osd:~  
GNU nano 2.3.1 File: fork.c  
#include<stdio.h>  
#include<sys/types.h>  
#include<unistd.h>  
  
int main()  
{  
    fork();  
    printf("Hello World\n");  
    return 0;  
}  
  
osd-190031187@team-osd:~  
[osd-190031187@team-osd ~]$ nano fork.c  
[osd-190031187@team-osd ~]$ gcc fork.c  
[osd-190031187@team-osd ~]$ ./a.out  
Hello World  
Hello World  
[osd-190031187@team-osd ~]$
```